ROBOFUN.RO

LECȚIA XIII

CURS GRATUIT

ARDUINO ȘI ROBOTICĂ

Mouse şi Tastatura cu Arduino Leonardo

Proiecte :

1. Logare presiune atmosferică, umiditate, nivel de iluminare şi temperatură în Excel, cu Arduino Leonardo
2. Mouse cu accelerometru şi Arduino Leonardo

Arduino Leonardo pe post de tastatura

Exact, daca ai un Arduino Leonardo, poti sa-l faci sa se comporte ca o tastatura sau un mouse absolut obisnuite (din punct de vedere al calculatorului). Este o facilitate a noului microcontroller Atmega32u4. Tot ce ai facut este ca in codul Arduino sa apelezi libraria Keyboard si calculatorul (la care ai conectat Arduino prin USB) se va comporta ca si cum ai fi apasat taste pe tastatura obisnuita.

Inainte de a trece mai departe, este necesar sa iti atrag atentia asupra unui lucru care ar putea fi neplacut. Imediat ce ai programat placa sa trimita taste apasate catre calculator, Arduino le va trimite tot timpul, pana cand ii spui sa se opreasca. Singurul mod in care ii poti spune sa se opreasca este incarcand alt program. Dar daca ai pus placa sa trimita taste prea rapid, atunci nu vei putea face acest lucru, pentru ca se vor apasa taste chiar in mediul de dezvoltare, acolo unde vei vrea sa scrii codul. Ca sa nu se intample asta, ai grija ca sa existe un mecanism prin care sa opresti trimiterea de taste (cum ar fi un buton conectat la Arduino, sau un delay mare pus un setup). Cu un delay in setup, poti oricand sa dai un reset placii altfel incat sa o fortezi sa ruleze functia setup, si apoi cat timp placa sta in delay, tu ii vei putea modifica programul.

Exemplu 1 -

```
void setup() {
      delay(15000);
      Keyboard.begin();
}

void loop() {
    Keyboard.print("Hello!");
    delay(10000);
}
```

In exemplul de mai sus, nu facem altceva decat sa initializam modul tastatura (Keyboard.begin()) si apoi sa trimitem la fiecare 10 secunde sirul de caractere "Hello" (exact ca si cum am apasa pe tastatura consecutiv tastele H e l l o. Am pus un delay in setup de 15 secunde in ideea ca atunci cand vei vrea sa schimbi programul placii, sa poti da un reset si astfel sa ai la dispozitie 15 secunde pentru a incarca noul program.

Cu Arduino Leonardo poti apasa si tastele CTRL, ALT, sau SHIFT, ca mai jos.

Exemplu 2 -

```
char ctrlKey = KEY_LEFT_CTRL;

void setup() {
  Keyboard.begin();
  delay(15000);
}

void loop() {
  delay(1000);
  Keyboard.press(ctrlKey);
  Keyboard.press('n');
  delay(100);
  Keyboard.releaseAll();
  delay(30000);
}
```

Dupa ce au trecut cele 15 secunde din setup, Arduino apasa tasta CTRL (si o mentine apasata), apoi apasa tasta "n" si apoi elibereaza ambele taste. Efectul, pe majoritatea sistemelor de operare, este deschiderea unei ferestre noi.

Mai jos, lista tuturor comenzilor asociate tastaturii emulate de Arduino.

Keyboard.begin()

- deschide o sesiune de emulare tastatura. Este necesar sa apelezi aceasta comanda inainte de a le putea folosi pe restul.

Keyboard.end()

- inchide o sesiune de emulare tastatura.

Keyboard.press(<TASTA>)

- apasa si mentine apasata o tasta (utila pentru a trimite combinatii de taste, gen CTRL + N).

Keyboard.print(<SIR_DE_TASTE>)

- apasa si elibereaza o tasta sau un sir de taste. Util pentru a apasa o tasta sau un sir de taste apasate succesiv spre calculator.

Keyboard.println(<SIR_DE_TASTE>)

- acelasi lucru ca print, doar ca la final apasa si tasta ENTER. Util pentru a introduce informatie intr-un document text si a trece automat la linie noua.

Keyboard.release(<TASTA>)

- elibereaza o tasta apasata anterior (cu Keyboard.press()).

Keyboard.releaseAll()

- elibereaza toate tastele apasate anterior (cu Keyboard.press())

Keyboard.write(<TASTA>)

–apasa si elibereaza o tasta.

Arduino Leonardo pe post de mouse

Exact in acelasi mod cum emuleaza tastatura, Arduino Leonardo poate fi vazut de calculator si ca mouse. Poate misca cursorul pe ecran la orice pozitie si poate simula apasarea pe butoanele mouse-ului.

Exemplu 1 -

```
void setup(){
  Mouse.begin();
}

void loop(){
    Mouse.move(50, 50, 0);
    delay(10000);
}
```

Codul de mai sus initializeaza modul mouse, iar apoi la fiecare 10 secunde misca pointer-ul mouse-ului cu 50 de pixeli pe fiecare axa.

Mai jos, lista tuturor comenzilor asociate controlului mouse-ului din Arduino Leonardo.

Mouse.begin()

–initializeaza modul de control al mouse-ului.

Mouse.end()

–inchide sesiunea de emulare mouse.

Mouse.click(<BUTON>)

- apasa si elibereaza un buton al mouse-ului. Valori posibile sunt MOUSE_LEFT , MOUSE_RIGHT, MOUSE_MIDDLE. Astfel, Mouse.click(MOUSE_LEFT) genereaza un click obisnuit pe butonul din stanga al mouse-ului.

Mouse.move(<X>, <Y>, <SCROLL>)

- genereaza o miscare a cursorului de mouse pe fiecare axa in parte. Primul parametru este miscarea pe axa orizontala, al doilea pe axa verticala, iar al treilea reprezinta deplasarea rotitei de scroll.

Mouse.press(<BUTON>)

- apasa (fara a elibera) unul dintre butoanele mouse-ului. Valori posibile sunt MOUSE_LEFT , MOUSE_RIGHT, MOUSE_MIDDLE. Astfel,

Mouse.press(MOUSE_LEFT) este ca si cum am apasa manual pe butonul din stanga al mouse-ului, fara a-l elibera.

Mouse.release()

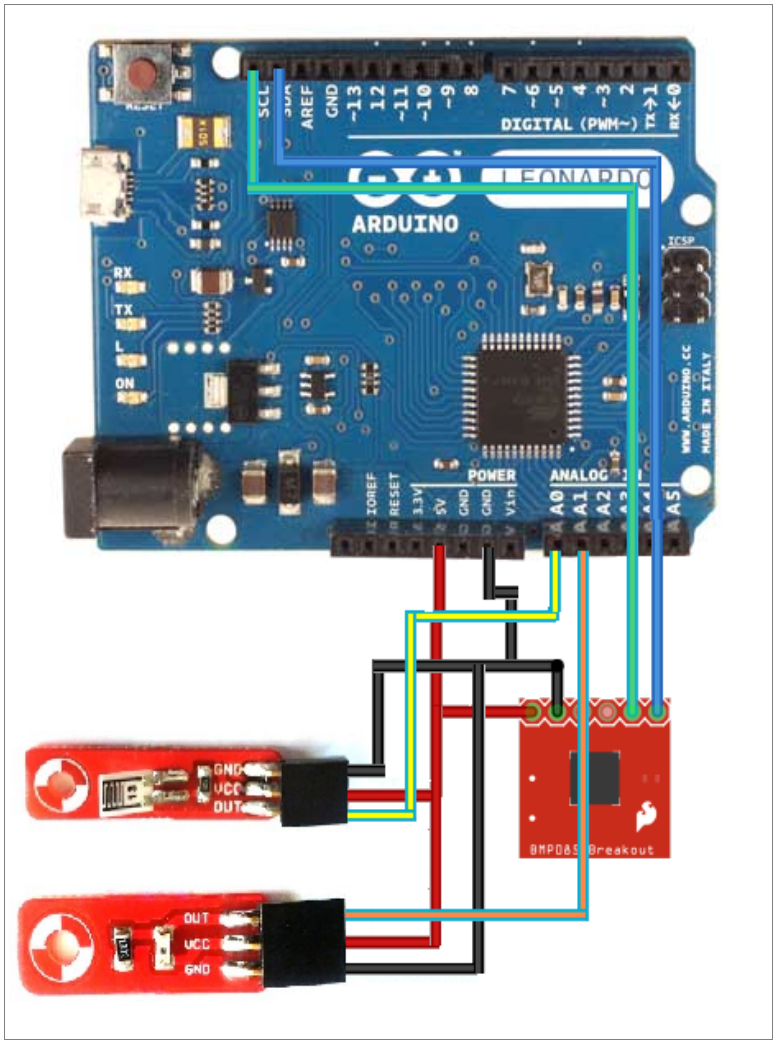
–elibereaza unul dintre butoanele mouse-ului apasate anterior cu Mouse.press. Valori posibile sunt MOUSE_LEFT , MOUSE_RIGHT, MOUSE_MIDDLE.

Logare presiune atmosferica, umiditate, nivel de iluminare si temperatura in Excel, cu Arduino Leonardo

Folosindu-ne de capacitatea lui Arduino Leonardo de a emula o tastatura, devine foarte simplu sa logam informatie pe calculator. Tot ce avem de facut este sa apasam din Arduino acele taste corespunzatoare valorilor citite (ca sa cand le-am introduce manual din tastatura).

Acest proiect foloseste un senzor BMP085 (presiune atmosferica si temperatura), un senzor de lumina brick si un senzor de umiditate brick, impreuna cu un Arduino Leonardo pentru a introduce periodic informatie intr-un document Excel.

Primul pas este conectarea componentelor la Arduino, ca in schema de mai jos. Probabil ca la prima vedere schema de conectare pare un fel de paianjen cu multe picioare, dar daca vei reveni la schema de conectare pentru fiecare senzor in parte (in sectiunea unde fiecare senzor este prezentat separat), atunci lucrurile vor deveni mult mai simple. Cele doua componente brick sunt conectate la 5V si GND pentru alimentare, iar apoi la pinii analogici A0 si A1. Senzorul BMP085 este conectat la cei doi pini I2C SDA si SCL (care in cazul Arduino Leonardo sunt in partea din stanga sus).


DIGITAL (PWM~)
ARDUINO
POWER
ANALOG IN
GND
VCC
OUT
OUT
VCC
GND

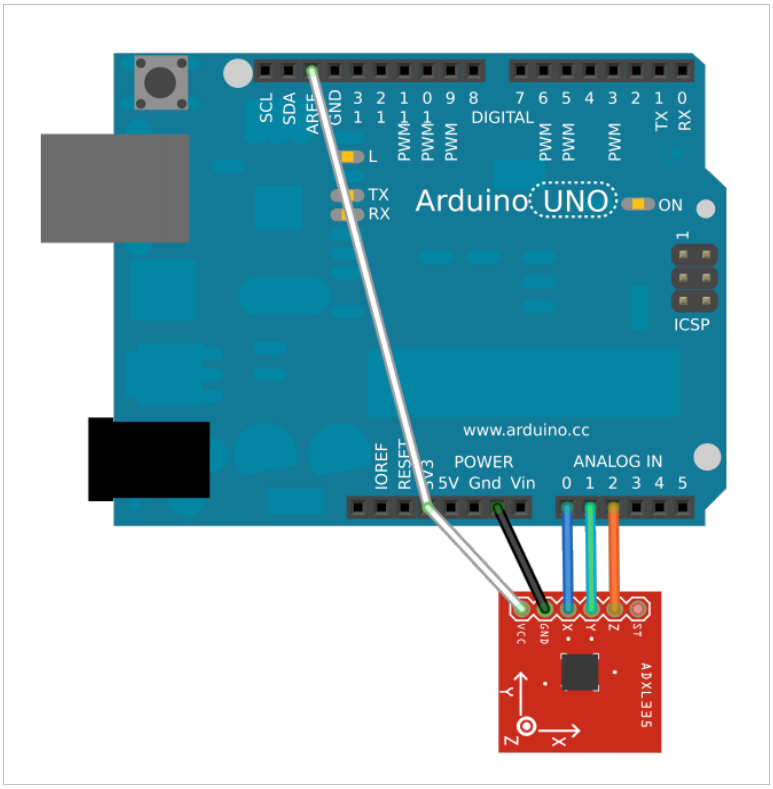
Mai departe, codul sursa nu face altceva decat sa combine exemplele care demonstreaza citirea datelor de la senzori cu cel care demostreaza utilizarea Arduino Leonardo pe post de tastatura.

```
#include <Wire.h>
#include <BMP085.h>

BMP085 dps = BMP085();
long Temperature = 0, Pressure = 0;

void setup(void) {
  Serial.begin(9600);
  Wire.begin();
  delay(1000);
  dps.init();
  delay(5000);
  Keyboard.begin();
}


void loop(void) {
  dps.getTemperature(&Temperature);
  dps.getPressure(&Pressure);
  int nivelIluminare = analogRead(1);
  int nivelUmiditate = analogRead(0);

  Keyboard.print(String(Temperature));
  Keyboard.print(KEY_RIGHT_ARROW);
  Keyboard.print(String(Pressure));
  Keyboard.print(KEY_RIGHT_ARROW);
  Keyboard.print(String(nivelIluminare));
  Keyboard.print(KEY_RIGHT_ARROW);
  Keyboard.print(String(nivelUmiditate));
  Keyboard.println();
}
```

Mouse cu accelerometru si Arduino Leonardo

Acest proiect se bazeaza pe capacitatea placii Arduino Leonardo de a se comporta ca mouse. Prin combinarea unui Arduino Leonardo cu un accelerometru pe 3 axe, vom obtine un mouse pe care il vom controla prin miscarea mainii. Poti folosi orice fel de accelerometrul doresti. In cele ce urmeaza vom folosi un accelerometru ADXL335, pentru simplitate. Accelerometrul se conecteaza la Arduino in modul obisnuit, ca in schema mai jos.

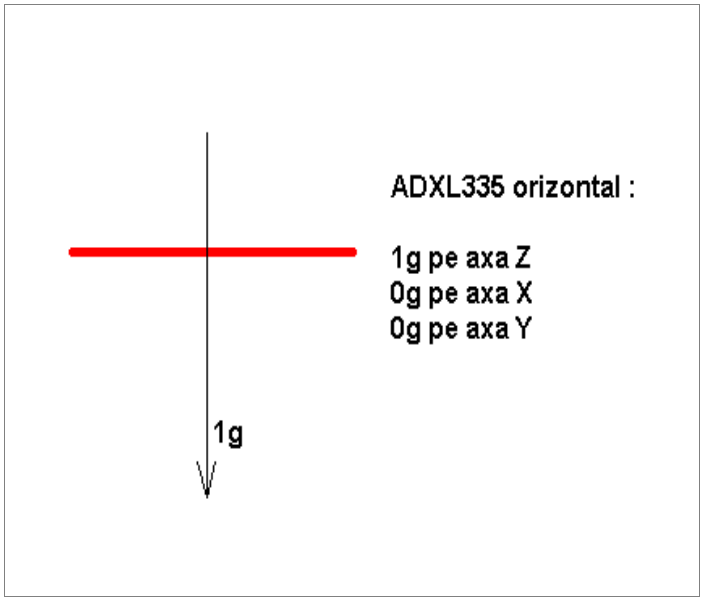
```
#define SENSIBILITATE_X 3
#define SENSIBILITATE_Y 3

void setup() {
 Serial.begin(9600);
 analogReference(EXTERNAL);
 Mouse.begin();

}

void loop() {
  float xAcc=readAcc(0);
  float yAcc=readAcc(1);
  float zAcc=readAcc(2);

  Serial.print("Acceleratie X: ");
  Serial.print(xAcc,DEC);
  Serial.print("Acceleratie Y: ");
  Serial.print(yAcc,DEC);
  Serial.println();
  delay(50);

  Mouse.move(xAcc * SENSIBILITATE_X, yAcc * SENSIBILITATE_Y, 0);

}

float readAcc(int port) {
 int value=analogRead(port);
 int miliVolts=map(value,0,1023,0,3300)-3300/2;
 float acc=(float)miliVolts/360;
 return acc;
}
```

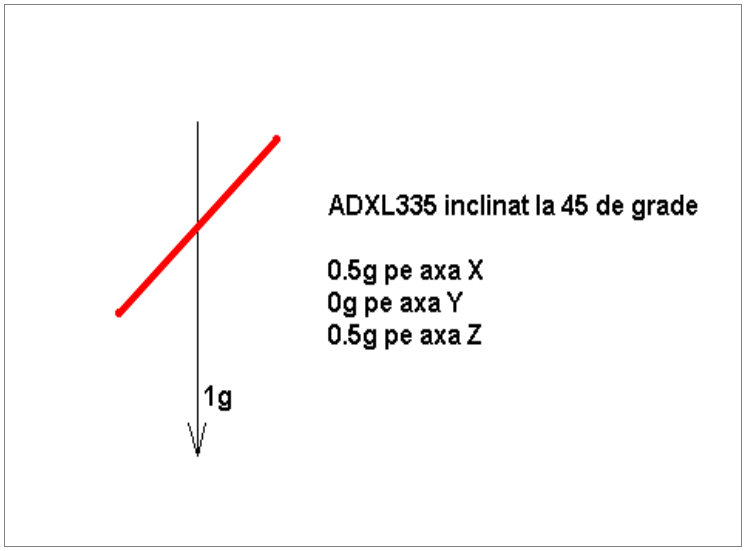


Arduino 3.3 V	ADXL335 VCC
Arduino GND	ADXL335 GND
Arduino Analog0	ADXL335 X
Arduino Analog1	ADXL335 Y
Arduino Analog2	ADXL335 Z
Arduino 3.3	Arduino AREF

Constantele SENSIBILITATE_X si SENSIBILITATE_Y definesc cat de mult se misca mouse-ul pe ecran atunci cand miscam accelerometrul. Cu cat valorile pentru aceste constante sunt mai mari, cu atat mouse-ul se va misca mai mult pe ecran atunci cand miscam accelerometrul. Esti liber sa te joci cu aceste constante pana cand miscarea pe ecran este exact asa cum vrei tu.

Principiul de functionare al acestui mouse se bazeaza pe faptul ca in orice locatie de pe Pamant avem o acceleratie gravitionala 1g, indreptata intotdeauna pe directia verticala. Cand accelerometrul ADXL335 este orientat pe directie orizontala, atunci pe axa X si pe axa Y acceleratiile citite sunt zero. Este o acceleratie de 1g citita pe axa Z. Atunci cand inclinam accelerometrul in plan, acceleratia de 1g care se manifesta doar pe axa Z cand accelerometrul este orizontal, incepe sa se manifeste si pe axele X si Y. Astfel daca inclinam accelerometrul la un unghi de 45 de grade pe axa X, vom avea o acceleratie de 0.5g pe axa X.


ADXL335 orizontal :
1g pe axa Z
0g pe axa X
0g pe axa Y
1g



Acceleratiile citite de accelerometru sunt inmultite cu o valoare constanta (in cazul nostru cu 3, valoarea constantei SENSIBILITATE_X) si mouse-ul este miscat pe ecran corespunzator valorii obtinute. Poate vei spune ca o deplasare de 3 pixeli atunci cand accelerometrul este inclinat maximum este putin, dar adu-ti aminte ca functia *loop* se executa non-stop, astfel incat mouse-ul este miscat cu 3 pixeli la fiecare 50 de milisecunde (pentru ca in *loop* exista o instructiune *delay(50)*).

Daca ti se pare complicat sa tii mouse-ul fix pe ecran (pentru ca este destul de dificil sa tii ADXL335 perfect orizontal), atunci putem imbunatati programul de mai sus prin setarea unei zone in care mouse-ul nu se misca deloc pe ecran, ca mai jos (modificarile in **bold**) :

```
#define SENSIBILITATE_X 3
#define SENSIBILITATE_Y 3
#define ZONA_MOARTA_X 0.3
#define ZONA_MOARTA_Y 0.3

void setup() {
 Serial.begin(9600);
 analogReference(EXTERNAL);
 Mouse.begin();
}

void loop() {
  float xAcc=readAcc(0);
  float yAcc=readAcc(1);
  float zAcc=readAcc(2);

  Serial.print("Acceleratie X: ");
  Serial.print(xAcc,DEC);
  Serial.print("Acceleratie Y: ");
  Serial.print(yAcc,DEC);
  Serial.println();
  delay(50);
```

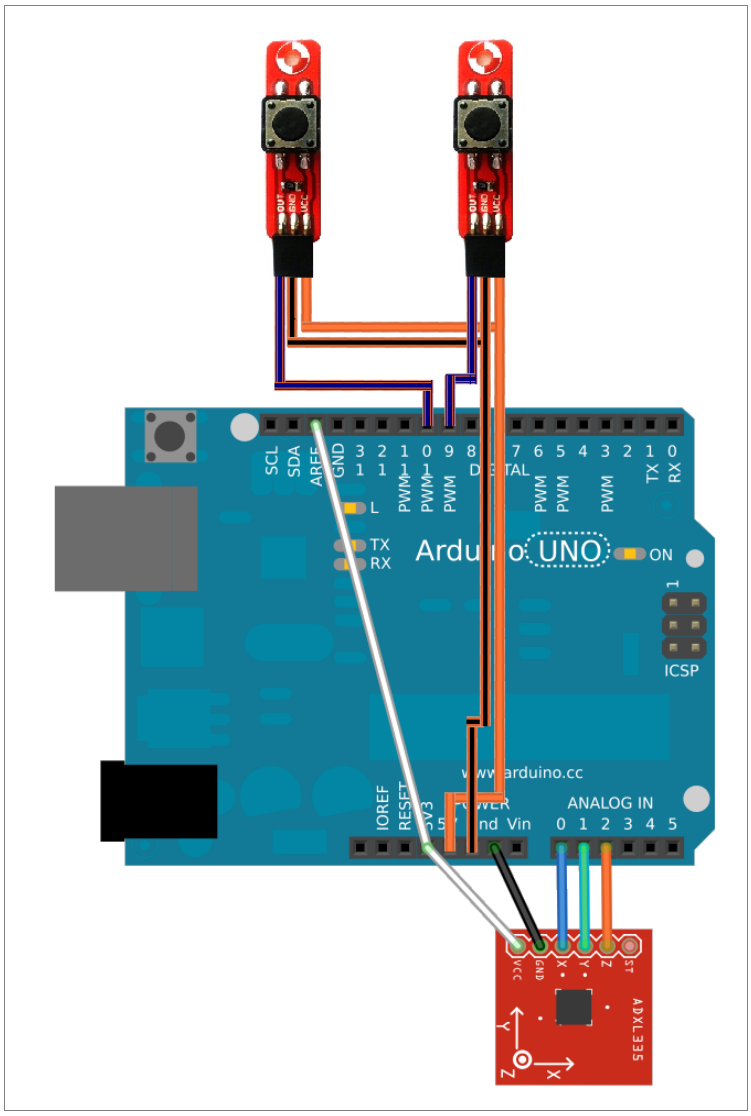
```
  int miscareX = 0;
  if (abs(xAcc) > ZONA_MOARTA_X) {
      miscareX = xAcc * SENSIBILITATE_X;
  }

  int miscareY = 0;
  if (abs(yAcc) > ZONA_MOARTA_Y) {
      miscareY = yAcc * SENSIBILITATE_Y;
  }

  Mouse.move(miscareX, miscareY, 0);

}

float readAcc(int port) {
 int value=analogRead(port);
 int miliVolts=map(value,0,1023,0,3300)-3300/2;
 float acc=(float)miliVolts/360;
 return acc;
}
```

Am introdus doua constante, ZONA_MOARTA_X si ZONA_MOARTA_Y, care definesc valorile acceleratiei sub care mouse-ul nu face nici o miscare. Astfel, daca tii accelerometrul intr-o pozitie aproape de orizontala, mouse-ul nu se misca deloc pe ecran.

In final, ca sa poti misca simplu accelerometrul, iti sugerez sa il prinzi pe o manusa de piele (cumparata de la standul cu elemente de protectie din Hornbach, BricoStore sau Practiker).

Urmatoarea extindere a acestui proiect este sa adaugi sistemului si doua butoane brick, obtinand astfel un mouse perfect functional. Schema de conectare este cea de mai jos. Butoanele le poti monta pe cate un deget al fiecarei manusi, astfel incat sa le poti apasa cu degetul mare (prin suprapunerea degetului mare peste unul dintre cele doua degete pe care sunt lipite butoanele).

```
#define SENSIBILITATE_X 3
#define SENSIBILITATE_Y 3
#define ZONA_MOARTA_X 0.3
#define ZONA_MOARTA_Y 0.3
```




```
#define BUTON1_PIN 9
#define BUTON1_PIN 10
#define DEBOUNCE_TIME 100

void setup() {
 Serial.begin(9600);
 analogReference(EXTERNAL);
 pinMode(BUTON1_PIN, INPUT);
 pinMode(BUTON2_PIN, INPUT);
```

```
 Mouse.begin();
}

long lastButtonPress1 = 0;
long lastButtonPress2 = 0;

void loop() {

  int buton1 = digitalRead(BUTON1_PIN);
  int buton2 = digitalRead(BUTON2_PIN);
  if (buton1 == 1) {
      if ((millis() - lastButtonPress1) > DEBOUNCE_TIME) {
            lastButtonPress1 = millis();
            Mouse.click(BUTTON_LEFT);
      }
  }
 if (buton2 == 1) {
      if ((millis() - lastButtonPress2) > DEBOUNCE_TIME) {
            lastButtonPress2 = millis();
            Mouse.click(BUTTON_RIGHT);
      }
  }

  float xAcc=readAcc(0);
  float yAcc=readAcc(1);
  float zAcc=readAcc(2);

  Serial.print("Acceleratie X: ");
  Serial.print(xAcc,DEC);
  Serial.print("Acceleratie Y: ");
  Serial.print(yAcc,DEC);
  Serial.println();
  delay(50);

  int miscareX = 0;
  if (abs(xAcc) > ZONA_MOARTA_X) {
      miscareX = xAcc * SENSIBILITATE_X;
  }

  int miscareY = 0;
  if (abs(yAcc) > ZONA_MOARTA_Y) {
      miscareY = yAcc * SENSIBILITATE_Y;
  }

  Mouse.move(miscareX, miscareY, 0);

}

float readAcc(int port) {
 int value=analogRead(port);
 int miliVolts=map(value,0,1023,0,3300)-3300/2;
 float acc=(float)miliVolts/360;
 return acc;
}
```

De remarcat in codul de mai sus este constanta DEBOUCE_TIME. Asa cum

am povestit mai pe larg in cadrul sectiunii despre butoane brick, atunci cand se apasa un buton, semnalul receptionat de catre Arduino variaza de cateva ori intre 0 si 1 (nu trece instantaneu din 0 in 1). Acest lucru se intampla datorita faptului ca atunci cand apasam butonul, lamelele metalice care inchid contactul nu se ating perfect. Ca sa evitam ca Arduino sa citeasca mai multe apasari de buton, vreme de 100 de milisecunde dupa prima apasare receptionata (valoarea lui DEBOUNCE_TIME) vom ignora orice alta apasare. Acest lucru inca permite conceptul de dublu click (doar ca diferenta intre doua click-uri succesive va fi exact DEBOUNCE_TIME milisecunde). Daca dublu click nu este sesizat corect de calculatorul tau, poti sa micsorezi valoarea acestei constante, sau sa modifici pe calculator durata intre cele doua click-uri succesive pentru a obtine un dublu click.

Aceasta a fost lectia 13. In final, as vrea sa te rog sa ne oferi feedback asupra acestei lectii, pentru a ne permite sa le facem mai bune pe urmatoarele.

Este vorba despre un sondaj cu 4 intrebari (oricare este optionala), pe care il poti accesa dand click aici.

Sau ne poti contacta direct prin email la contact@robofun.ro .

Iti multumim,

Echipa Robofun.RO